의료영상 회의시스템을 위한 CORBA기반의 서버시스템의 설계 및 구현

박세명[†] · 강재효[††] · 김상균[†††] · 최항묵[††††] · 최흥국[††††]

요 약

CORBA가 명세하는 객체참조 메커니즘인 이름 서비스와 트레이딩 서비스만으로는 MICS시스템이 요구하는 다양한 기능을 수용할 수 없으므로 추가적인 객체관리자의 구현이 필요하다. 따라서 본 논문에서는 MICS 시스템에서 영상처리 객체의 부하와 결함과 관련된 실시간 정보를 이용한 부하분배와 결함허용을 제공하는 객체참조서비스를 제공하며, 영상처리객체의 관리와 사용자환경의 동적 구성을 지원할 수 있는 객체관리자를 설계, 구현하였다. 제안된 객체관리자는 이질의 분산 환경에서 MICS시스템의 신뢰성과 확장성을 제공하여 클라이언트와 서버간의 고품질의 서비스를 제공한다.

Design and Implementation of Server System for MICS(Medical-Image Conference System) based on CORBA

S.M. Park[†], J.H. Kang[††], S.K. Kim[†††], H.M. Choi[††††] and H.K. Choi[††††]

ABSTRACT

According to CORBA specification, there are some mechanisms for object reference management such as Naming Service and Trading service, but these mechanism can't supply all functionalities needed fomanagement in MICS. So, We have designed and implemented additional special-purpose IPO manager. It provides load balancing service based on real-time information for IPO and fault-tolerable services with the management of redundant IPOs and slso provides transparent configuration of client environment with the lists of registered IPOs.

1. 서 론

메인프레임을 근간으로 하는 중앙 집중형 정보 시스템이 가지는 고비용, 저 효율의 문제점을 해결하고자 80년대 초기부터 분산처리에 관한 연구가 활발히 이루어졌고, 90년대에 들어 네트워크의 보급 확대로 인해 Client/Server 컴퓨팅 환경이 대두되었으며, 다양한 시스템에 대하여 단일의 인터페이스를 통해 접근이 가능하여 유지 보수에 대한 비용 절감을 이룩할 수 있으며, 시스템의 구조적 추상화가 가능하여 많은 유연성을 얻을 수 있는[1].미들웨어방식이 제시되었다. 이러한 움직임은 객체지향 소프트웨어공학 기술과 인터넷이라는 개방형 컴퓨팅 환경이 어우러져 분산객체(Distributed Objects) 기술로 집약되고 있다[5].

분산객체 기술을 이용한 분산응용 개발과정에서 대두되는 사용언어와 플랫폼 독립성 문제의 해결을 위해 세계 각 국의 700여개 기업체 및 연구소등의 컨소시움인 OMG(Object Management Group)에 의해 분산객체 기술 표준안인 CORBA(Common Object Request Broker Architecture)[3,4]가 제시되게 되었다. CORBA는 클라이언트들에게 서비스 제공자들의

본 연구는 1998년 인제장학재단의 연구비지원에 의한 것임.
† 준회원, 인제대학교 정보컴퓨터공학부 부교수
†† 준회원, 에이블컴(정보기술팀)
††† 정회원, 인제대학교 정보컴퓨터공학부 조교수
†††† 종신회원, 인제대학교 정보컴퓨터공학부 조교수

위치에 대한 투명성을 보장해 줄 수 있는 객체참조 메커니즘으로 이름 서비스(Naming service)와 트레이딩 서비스(Trading Service)를 명세하고 있다. 또한 이름 서비스와 트레이딩 서비스의 효율적인 서비스제공을 위한 연구가 진행된 바 있으나[12,18] 다양한 응용분야에서 분산객체 시스템의 적용을 고려해 볼 때 ORBA 명세서에서 고려하고 있는 이름 서비스나 트레이딩 서비스만으로는 각 응용분야에서 요구하는 객체참조기능을 충족시킬 수 없다.

본 논문에서 고려한 MICS시스템에서는 다양한 영상처리 기법을 가능케 하는 영상처리객체의 제공을 위한 영상처리 객체의 수정, 추가, 삭제등의 영상처리객체의 관리기능, 영상처리요청에 대한 적절한 작업반환시간을 위한 동적인 부하 분산기능, 영상처리 객체의 결합허용 기능, 그리고 사용 가능한 영상처리 객체의 종류에 무관하게 사용자 인터페이스의 동적 구성을 지원할 수 있는 Metadata[18]설계패턴(design pattern)이 요구되나 현재 CORBA 명세서에 기술된 객체참조 서비스로는 이러한 요구를 충족시키지 못한다.

따라서 본 논문에서는 MICS시스템을 구성하는 객체들 중에서 영상처리 객체의 효율적인 관리를 위한 부가적인 기능을 갖는 객체관리자를 구현하였다. 본 논문에서 설계, 구현한 영상처리객체 관리자는 영상처리 객체의 부하와 결함과 관련된 실시간 정보를 이용한 부하분배와 결함허용을 제공하는 객체참조서비스를 제공하며, 영상처리객체의 관리와 사용자 인터페이스의 동적 구성을 지원할 수 있는 목록정보를 유지한다. 구현된 객체관리자는 이질의 분산 환경에서 MICS시스템의 신뢰성과 확장성을 제공하여 클라이언트와 서버간에 고품질의 서비스를 제공한다.

본 논문의 구성은 다음과 같다. 2장에서는 MICS 시스템을 위한 영상처리객체 관리자의 필요성을 제시하고, 3장에서는 객체관리자(Object Manager)를 설계하였으며, 4장에서는 설계된 객체관리자를 MICS [6-8] 시스템에서 구현한 예를 보이며, 마지막 결론에서는 앞으로 객체관리자에서 추후 연구과제에 대해서 기술한다.

2. MICS시스템의 영상처리객체 관리자의 필요성

MICS시스템은 시스템의 확장에 따라 객체의 종류와 수는 더욱 증가되므로 사용자의 요구에 따라 서비스가 가능한 객체참조의 획득 여부가 시스템의 성능을 좌우하는 중요한 요소이다. 현재 CORBA 명세서에 규정한 객체관리 방법은 크게 이름서비스와 트레이딩서비스를 들 수 있다. 이름서비스는 등록을 요구하는 서비스 객체의 위치 정보, 즉 참조를 유지하고 있으면서 클라이언트의 요구가 있을 때, 클라이언트로부터 넘겨받은 이름 값을 기반으로 해당하는 서비스 객체를 검색하여 클라이언트에게 객체참조를 넘겨주도록 설계되었다. 이러한 이름-참조 값 기반의 검색 방법은 클라이언트에게 사용하고자 하는 서비스 객체의 이름 정보를 정확히 알고 있어야 하기 때문에 특정 서비스객체를 사용하는 클라이언트는 해당 객체의 이름을 가지고 있어야 하며, 이는 서비스 객체의 추가, 삭제, 그리고 수정에 따라 클라이언트코드도 변경되어야 함을 의미한다. 또한 일부 제품의 이름서비스는 참조가 요청된 이름의 객체가 여러개 등록된 경우에는 라운드 로빈 방식으로 등록된 순서대로 객체의 참조를 반환하는 단순한 부하분산의 기능을 가지고 있으나 처리객체의 실행과 관련된 동적인 정보는 전혀 고려하지 못하고 있다. 트레이딩 서비스를 이용하면 객체의 등록된 이름을 사용함으로써 발생할 수 있는 문제점은 방지할 수 있으나 새로운 영상처리객체가 추가되는 경우 클라이언트의 수정을 피할 수 없으며, 부하분산을 위한 메커니즘 역시 제공하지 못하고 있다.

이처럼 기존의 객체관리 기법은 찾고자 하는 객체의 이름 또는 속성에 대응하는 객체의 참조만을 반환하도록 설계되었으므로 응용목적에 적합한 부가기능이 첨가된 객체참조 기능을 갖는 별도의 객체관리자가 필요하다. 그러므로 시스템을 구성하는 객체들 중에서 단순히 이름 또는 속성 값에 해당하는 객체의 참조만이 요구되는 경우 즉, 한번 구현되면 변화지 않는 객체들의 참조관리를 위해서는 다른 시스템과의 호환성 및 개발의 용이성을 위해 기존의 서비스를 사용하는 것이 바람직하나 해당 시스템만의 고유한 처리를 위한 객체들 즉, 시스템의 발전에 따라 추가 및 삭제가 발생할 수 있는 처리객체들을 위해서는 처리객체의 응용목적에 적합한 부가기능을 갖는 객체관리 방안이 새로 제시되어야 한다.

3. 영상처리객체 관리자 설계

본 논문에서 설계 및 구현하고자 하는 서버시스템

의 핵심 요소인 MICS시스템의 영상처리객체 관리자에서 고려하는 결함허용 및 부하분산 정책, 클라이언트 동적 구성 지원 그리고 객체관리자의 구성에 대한 설명은 다음과 같다.

3.1 결함허용 및 부하분산

결함허용 시스템이라 함은 네트웍의 붕괴나, 머신의 다운, 그리고 처리객체의 비정상적인 상태 등에 의한 서비스의 불완전성이나 불가능성에 대한 충분한 고려가 이루어진 시스템으로[1] 본 논문에서는 영상처리객체 관리자에 의해 관리되는 영상처리객체의 결함허용에 대해 고려하였다. 처리객체의 결함허용을 위해서 주로 처리객체의 복제기법을 통한 해결책이 이용되고 있다. 즉, 서로 상태가 같은 처리객체를 여러 곳에 분산시켜 두었다가 어느 한 곳의 처리객체가 서비스를 수행하지 못할 경우 다른 처리객체가 서비스를 제공하도록 함으로써 클라이언트에게 서비스 제공에 대한 신뢰성을 보장해주는 메커니즘으로[15] 복제방식으로는 Primary backup replication과 Active replication을 들 수 있다[2,15,16].

본 연구에서의 결함허용 방식은 각 처리서버에 처리객체들을 각각 준비해두고, 처리객체와 처리서버에 대한 정보를 바탕으로 적절한 처리객체 중에서 하나를 선택하도록 함으로써 서비스의 불가능성을 방지하도록 한다는 점에서 Primary backup replication 방식과 차별되며, 각 서버에 동일한 처리객체들이 준비되어 있다는 점에서는 Active replication과 유사하나 모든 객체가 동일한 작업을 동시에 수행하지 않는 다는 점에서 Active replication과는 다르다. 본 논문에서 구현한 MICS시스템은 응용시스템이기 때문에 특정 회사의 ORB에 종속되는 것은 바람직하지 않으므로 가능한 기존 ORB에서 제공하는 객체 활성화 기능을 사용하도록 하였다. 각 영상처리객체는 각 처리서버에 등록되어 서비스 가능 상태로 있게되며, 사용자의 요구가 특정 서버의 복제객체에게 전달되면 그때 해당 객체가 활성화되어 요구를 처리한다. 이때 어느 서버에게 요구를 전달할 것인가는 영상처리객체 관리자의 객체 모니터링 기능에 의해 수집되어진 실시간 정보들을 바탕으로 사용자의 요청에 대한 최적의 서비스가 가능한 객체를 선택하게 하였다. 그러나 본 논문에서는 요청이 전달된 후 처리되지 못하는 경우의 오류처리 방안은 포함하고 있지 못하다는 점에서 보완이 필요하다.

분산객체 시스템에서 현재 연구되고 있는 부하분산에 대한 접근 방법으로 그룹객체(Object- Group) 모델을 들 수 있으며, 최근에 이러한 그룹 모델에 대한 새로운 정의를 내리기 위한 연구에서는 객체그룹은 응용그룹(application group)과 복제그룹(replication group)으로 두 가지 유형으로 나누고 있다[2]. 응용그룹은 서로 다른 서비스를 제공하는 객체들의 그룹을 지칭하며, 복제그룹은 동일한 기능을 수행하는 복제된 객체들의 그룹을 의미한다. 본 논문에서는 사용가능한 처리서버별로 임의의 서비스를 위한 복제된 객체가 존재하도록 구성하였다. 즉 각 처리서버에는 응용그룹에 속하는 모든 객체를 설치하고, 영상처리객체 관리자는 사용자의 요청이 발생하면 이를 어느 처리서버에서 처리할 것인지를 결정하게 함으로써 부하분산을 시도하였다.

본 논문에서는 영상처리객체를 처리할 처리서버들이 별도의 통신망에 연결되어 있으므로 부하분산을 위한 기준을 선정함에 있어 네트워크 비용 조건은 모든 처리서버가 동일하다고 고려하였으며, 또한 현재 구현된 시스템은 3개의 유사한 하드웨어 사양을 가지는 처리서버로 구성되어 있으므로 처리서버의 성능인자는 고려하지 않았다. 따라서 본 논문에서는 부하분산을 위한 기준 인자로 요청된 처리객체의 사용가능 여부와 각 처리서버별로 처리중이거나 대기중인 작업의 수를 사용하였다. 이를 위해 처리서버의 각 처리객체의 사용가능 여부를 판단하기 위해서 플래그(Busy Flag)를 이용하였으며, 각 처리서버 별로 처리중이거나 대기중인 작업의 수를 위해 작업 계수기(Work Counter)를 사용하였다. 영상처리객체 관리자는 수시로 각 영상처리객체의 상태를 파악하여 사용가능 여부를 판정하고, 클라이언트의 요청이 발생하면, 그 시점에서 사용가능하며 처리할 작업이 가장 작은 처리서버를 선정하여 해당 서비스 객체에게 작업을 의뢰하여 각 서버의 작업 계수기의 값을 비슷한 수준으로 유지하는 정책을 수행함으로써 부하분산을 시도하였다. 그러나 본 논문에서 구현하고자 하는 영상처리객체 관리자의 경우 작업대상 영상의 크기가 사전에 알려지는 특징이 있어 각 영상처리객체가 해당 영상을 처리하는 시간을 어느 정도 예측가능하며, 분산시스템의 특성상 서로 다른 성능을 갖는 처리서버가 존재할 수 있으므로 이러한 작업처리시

간과 처리서버의 성능 인자를 부하분산 정책에 포함시키는 연구가 진행중이다.

3.2 클라이언트 동적 구성 지원

영상처리객체 관리자는 영상처리객체가 등록될 때 획득한 응용그룹 이름과 복제그룹 이름을 이용하여 목록을 구성하고, 새로운 영상처리객체가 등록되어 목록이 갱신되면 이를 회의관리자에게 전송한다. 회의관리자는 항상 최신의 목록을 유지하면서, 사용자가 접속할 때 영상처리객체의 목록과 사용자인터페이스 프로그램을 전송하게되고, 사용자 인터페이스 프로그램은 함께 받은 목록을 이용하여 메뉴기반의 사용자 환경을 초기화시킨다. 사용자환경 초기화시에 특정 영상처리객체의 응용그룹 이름은 사용자 환경에서 주메뉴 항목으로 등록되며, 복제그룹 이름은 주메뉴의 부 메뉴 항목으로 표시된다. 사용자 환경에서 메뉴항목으로 표시되는 영상처리객체의 그룹이름은 영상처리객체의 기능을 표현할 수 있으므로 사용자는 메뉴의 기능을 쉽게 알 수 있으며 주메뉴의 특정 서브메뉴를 선택함으로써 선택된 영상에 대한 필요한 처리를 요청할 수 있다. 이처럼 영상처리객체 관리자가 등록된 영상처리객체의 이름을 관리, 배포함으로써 사용자 인터페이스 프로그램의 영상처리 요청 인터페이스를 Dynamic Attribute 설계 패턴[18]을 이용하여 단일화할 수 있어, 새로운 객체의 등록이나 객체의 삭제가 발생하는 등 시스템이 변화하더라도 기존의 사용자 인터페이스 프로그램에 어떠한 수정도 없이 사용되어질 수 있다.

3.3 영상처리객체 관리자 구성 및 각 모듈의 동작

영상처리객체 관리자는 크게 3개의 모듈로 구성되며 구성도는 그림 1)와 같다. 3개의 모듈중 검색(Lookup)모듈과 등록(Register)모듈은 회의관리자를 위한 인터페이스를 가지며, 객체감시(Object Monitor) 모듈은 영상처리객체의 인터페이스를 사용한다. 등록 모듈은 등록된 객체들의 참조에 관련정보(그룹이름, 객체참조)를 유지하고, 객체감시 모듈은 등록 모듈에 의해 생성된 정보를 이용하여 처리객체가 등록된 서버정보와 각 처리객체의 실시간 정보를 생성, 유지한다. 영상처리객체 관리자와 회의관리자(클라이언트), 그리고 서비스객체간의 상호작용에

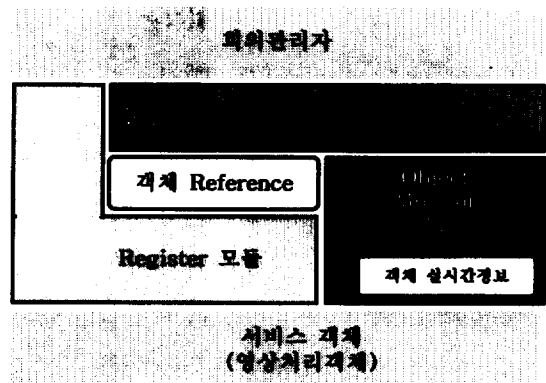



그림 1. 객체관리자 구성도

대한 설명은 다음과 같다.

서비스객체는 각 처리서버에서 객체관리자의 등록 모듈 인터페이스를 통해 응용그룹 이름과 복제그룹 이름을 등록한다. 이때 Register 모듈은 등록시 제시된 이름에 따라 새로이 등록될 객체가 새로운 서비스 객체이면 응용그룹에 참가시키고, 기 등록된 객체의 복사본이면 응용그룹의 하위 그룹으로 등록된다. 등록모듈은 새로운 서비스 객체가 등록되거나 객체감시 모듈로부터 더 이상 사용가능하지 않다는 통지가 오면 사용 가능한 그룹이름만 가지는 서비스 목록을 재구성하고 이를 회의관리자에게 전송한다. 회의관리자는 개설된 모든 회의가 종료되면 서비스 목록을 갱신한다. 클라이언트는 접속 시 회의관리자로부터 서비스 목록을 내려 받아 메뉴를 동적으로 구성한다. 객체감시 모듈은 등록된 서비스 객체의 상태를 체크하여 처리서버별 해당 서비스 객체들의 상태를 기록, 유지한다. 더 이상 사용가능하지 않은 서비스 객체에 대한 정보를 등록모듈에 통지하며, 시스템관리자는 해당 정보를 검색하여 봄으로써 객체들의 상태를 파악할 수 있다.

클라이언트의 서비스 객체에 대한 서비스 요구가 회의관리자를 통해 객체관리자의 검색 모듈 인터페이스를 통해 넘겨지면, 검색모듈에서는 요구되어진 서비스를 해석하여 객체감시 모듈로 의뢰를 전달한다. 객체감시 모듈은 서비스 객체들의 상태를 참조하여 최적의 서비스 제공이 가능한 서비스 객체와의 연결을 설정한 후, 해당 처리서버의 작업계수기를 증가시키고 해당 처리객체의 Busy Flag를 True로 설정한다. 서비스 객체의 처리 작업이 완료되어 결과

값을 반환하고 해당 처리서버의 작업계수기 값을 감소시키고, 해당 처리객체의 Busy Flag는 다시 False로 전환한다. 각 모듈에 대한 상세한 설명은 다음과 같다.

3.3.1 Register 모듈

등록모듈을 통해서 새로운 서비스 객체가 등록되거나 객체감시모듈로부터 더 이상 사용가능하지 않은 객체에 대한 통지가 오면 이를 서비스객체 목록에서 삭제하고 클라이언트의 초기화를 위한 서비스객체 목록을 재구성하여 회의관리자에게 전달한다. 서비스 객체는 자신의 서비스 계보를 인수로 하여 시스템에 가입을 요청한다. 여기서 서비스 계보는 그룹을 형성하는 하위 그룹과 멤버를 계보화 시킨 것으로 등록모듈은 인수로 전달된 서비스 계보를 계보표를 유지하는 트리 구조에 삽입시킨다. 트리 형식의 계보도는 그룹이름만을 포함하는 리스트 형식으로 표현하여 파일로 저장한 후 회의관리자에게 전달되며, 클라이언트의 동적 메뉴 구성 시 사용된다. 그림 2)와 3)에서 트리 형식의 계보표와 리스트 형식의 계보표 표현 방식을 보여준다. 그림 2)의 계보표 중에서 실제 회의관리자에게 전달될 목록에는 처리객체에 대한 정보는 포함되지 않는다.

그림 2. 트리형식의 계보도

<SubGroup1<SubGroup1-1<처리객체 4>><처리객체 1>>
<SubGroup2<처리객체 2>>
<SubGroup3<SubGroup3-1<처리객체 5><처리객체 6>><처리객체3>>

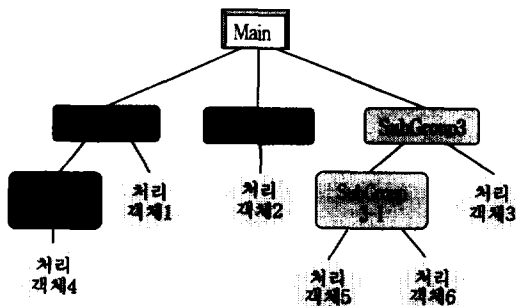
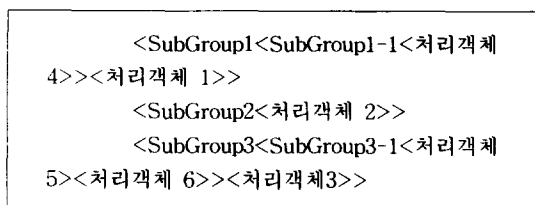



그림 3. 리스트 형식의 계보표

3.3.2 Object Monitor 모듈

객체감시 모듈은 등록된 객체들의 상태를 관리하는 역할을 담당한다. 즉, 현재 등록된 객체들이 어느 처리서버에 설치되어있으며, 사용 가능한지에 대한 정보를 생성, 유지한다. 이런 정보들을 이용하여 사용자의 요청을 적절한 처리객체에게 전달한다.

3.3.3 Lookup 모듈

검색모듈은 클라이언트와의 인터페이스를 이루며, 클라이언트로부터 전달받은 서비스 요구를 해석하여 사용자가 원하는 객체를 찾아 사용자의 요구가 전달되도록 한다. 이때 사용되어지는 객체의 이름은 등록모듈에서 사용되던 객체의 계보 이름과 같다. 따라서 계층적 구조를 가지고 관리되어지는 시스템의 그룹객체들에서 검색하고, 등록되어 있으면 객체감시모듈에게 의뢰하여 적절한 객체에서 전달할 것을 요청한다.

그림 4)는 객체관리자의 요청 처리 흐름을 보여주고 있다. 최초 클라이언트의 서비스 요청을 받았을 때는 서비스를 제공할 객체가 선택되어 있지 않으므로 검색모듈을 통해 서비스 가능한 객체의 레퍼런스를 찾는다. 만약 검색되어진 객체가 없다면 예외처리를 통해 서비스 불가한 요구를 주었다는 메시지를 클라이언트에게 전달하고 새로운 요구를 기다린다.

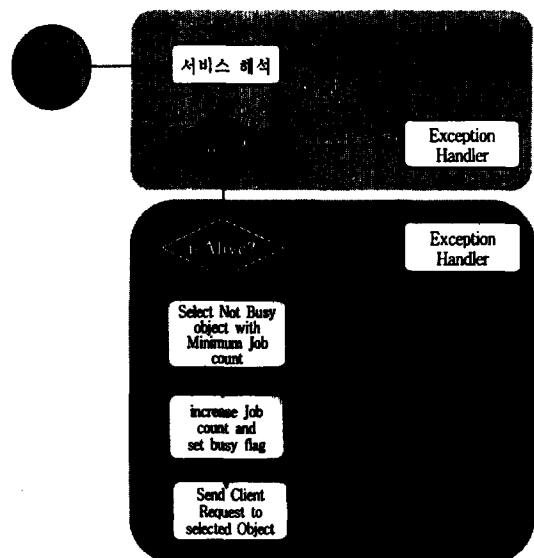



그림 4. Object Manager의 Request 처리 흐름도

정상적으로 하나 이상의 객체 참조를 검색하였다면 검색되어진 객체가 활성화되어 사용 가능한지 체크를 한다. 만약 운용이 불가한 상태라면 역시 예외처리를 통해 사용불가를 클라이언트에게 전달하며, 운용이 가능한 상태이면 현재 다른 서비스 요구를 처리하고 있는지를 파악한다. 이때 사용되지 않는 서비스 객체에 가중치를 높게 두어 선택 될 수 있는 기회를 높게 부여함으로써 부하분배를 유도할 수 있다.

4. MICS 시스템 구현 및 고찰

4.1 객체관리자의 구현 및 설치환경

본 논문에서 제시한 객체관리자를 의료영상 회의 시스템(MICS : Medical Image Conference System)에서 구현해 보았으며, 그림 5)는 MICS의 전체 구성도를 보여준다. 본 MICS의 구성요소중 사용자인터페이스와 회의관리기능, 그리고 IPO (Image Processing Object)에 대한 상세설명은 참고문헌을 참고하기 바라며, MICS시스템의 개략적인 동작은 다음과 같다.

Applet은 회의 개설자 Applet과 회의 참여자 Applet으로 구분된다. 회의 개설자는 초기 HTML문서에서 개설자로 등록하고 개설자 Applet을 다운 로드 받는다. 개설자 Applet은 회의 개설 메뉴와 영상처리 메뉴를 가진다. 회의 참가자는 HTML문서에서 현재 개설된 회의 목록을 보고, 회의를 선택하면, 참가자 Applet을 다운로드 하게된다. Applet은 실행과 동시

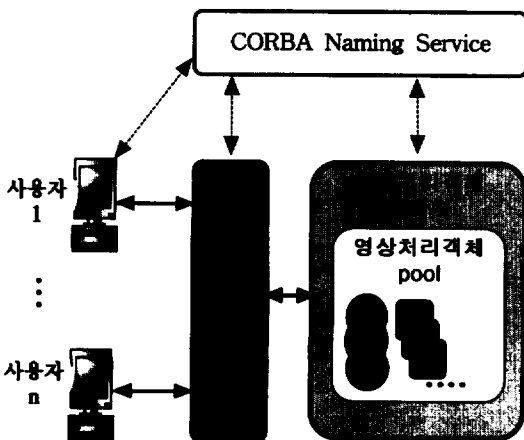



그림 5. MICS 시스템 구성도

에 회의관리자에게 자신의 Callback Object 참조를 전달하여, 이후 서버에 새로운 이벤트(메시지, Whiteboard 좌표, 처리된 영상 등)가 발생 시 서버가 능동적으로 클라이언트에게 이벤트를 전달 할 수 있도록 한다.

Conference Manger Object는 회의 개설자가 회의 개설을 신청했을 때 생성되며, 회의 참가자들의 객체 참조 정보를 유지하며, 클라이언트로부터 발생한 메시지나 Whiteboard 이벤트 정보를 회의 참가자 Applet에 전달하며, 개설자의 영상처리에 관한 요청은 객체관리자에게 전달하며, 처리결과를 회의 참가 사용자 Applet에게 전달한다.

객체관리자는 회의관리자로부터 영상처리 요청을 받아 영상처리객체에게 영상처리 요청을 하고 처리결과를 회의관리자에게 되돌려 주는 역할을 담당하며영상처리객체를 관리한다.

영상처리객체는 하나의 영상 처리 기능을 가진다. 객체관리자로부터 영상처리 요청을 받아 요청한 영상에 대한 영상처리를 한 다음 JPEG으로 압축하여, 영상처리객체에게 그 결과를 되돌려 준다. 'Visual C++'로 구현된 이 Object는 네트워크의 부담, 영상처리 부담을 고려해 하나의 머신(machine)이 아닌 여러 개로 분리 할 수 있다. 그림 6)은 객체관리자를 정의하는 IDL이다. 객체관리자의 외부 인터페이스는 서비스 객체를 위한 Reqister 인터페이스와 클라

```
module OManager {
    struct ImageInfo {
        Mics::ImageData image;
          string fileName;
        };
    typedef sequence<ImageInfo> ImageList;
    // from ConferenceManager to ObjectManager
    interface ObjectM {
    // 회의 개설시의 원본영상 전송
    oneway void sendImage( in Mics::ImageData
            image, in long len, in  string filename);
    // 영상처리 요청
    Mics::ImageData lookup(
        in string operationName, in string filename )
          raises(Mics::IPONotFound);
    // 영상처리 루틴 등록
    oneway void registerIPO( in long depth,
          in string name, in string lastName,
    in IPO::ImagePO objref );
    void getMenuInfo( out Mics::MenuFile menu );
};
```

그림 6. 영상처리객체 관리자의 IDL

이언트를 위한 Lookup 인터페이스 두 종류가 있다. 이 외에도 참조 관리자에서 관리되어지는 리스트 형식의 계보표를 얻기 위한 메소드인 getMenuInfo()와 클라이언트가 회의를 개설하였을 때 회의에 사용될 영상을 전달하기 위한 sendImage() 메소드가 정의되어 있다. 그림 7)은 객체관리자의 주요 클래스를 보여준다.

그림 8)은 이상에서 설명한 MICS시스템의 설치도를 나타낸다. 본 시스템에는 Web서버 및 회의관리자를 위해 Compaq 서버를 이용하고 있으며, 영상처리객체관리자와 영상처리객체를 위해 3대의 IBM PC(PentiumII 400Mhz)를 이용하였다. 사용자환경을 구축을 위해 가장 우선적으로 고려한 것은 접근성으로 인터넷을 사용할 수 있는 환경이면 별도의 클라

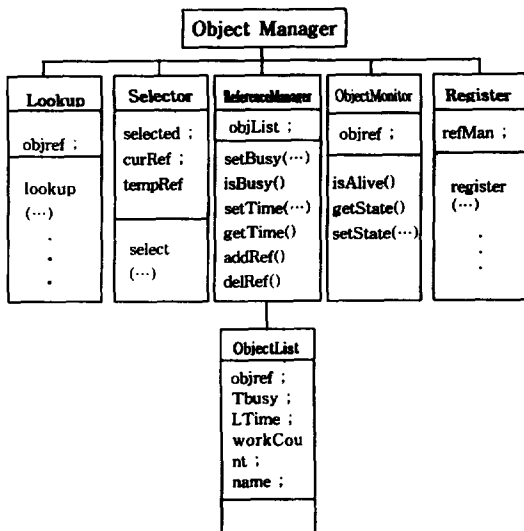



그림 7. 객체관리자의 주요 클래스

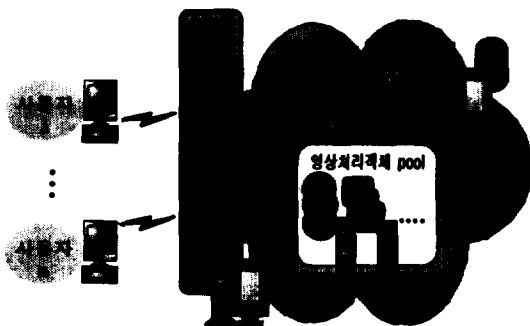



그림 8. MICS 시스템 설치도

이언트 프로그램을 설치하지 않고도 웹브라우저 상에서 동작할 수 있도록 하기 위해서는 Inprise사의 Java용 제품을 이용하여 Netscape상에 Applet으로 구현하였다. Netscape 웹브라우저는 자체로 Inprise사의 Java용 CORBA 제품과 호환되는 ORB를 포함하고 있으므로 Applet수행을 위해 필요한 코드를 내려 받는 시간을 단축할 수 있다. 회의관리자와 영상처리객체관리자 그리고 영상처리객체는 처리속도가 시스템에 중요한 영향을 끼치므로 C++ 용 CORBA 제품인 MICO를 이용하였다.

4.2 실행 결과 및 고찰

그림 9)는 Netscape 브라우저에서 Java Applet으로 구현된 사용자환경을 보이고 있다. 그림에서 좌측의 영상은 초기 영상이며, 우측의 영상은 처리결과 영상을 보이고 있으며, 우측의 결과 영상은 여러 가지 처리결과를 템메뉴를 사용하여 선택할 수 있도록 하고 있고, 좌측하단에는 문자기반의 의사전달도구를 보여주고 있다.

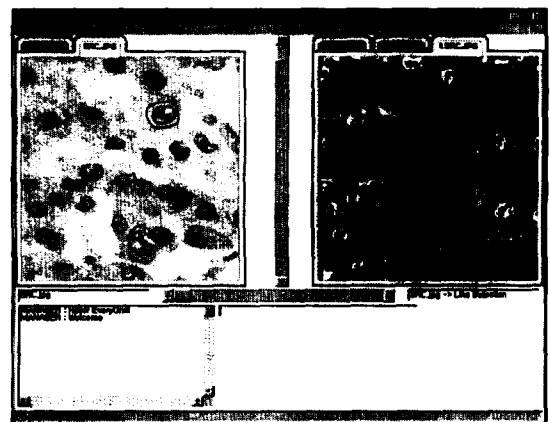



그림 9. MICS의 실행 화면

본 논문에서 구현한 MICS 시스템은 각 영상처리객체를 각 처리서버에 중복 배치함으로써 영상처리객체의 결합허용을 부여하였으나 영상처리 관리자와 회의관리자의 결합허용은 고려하지 못하고 있으므로 이에 대한 보완도 요구된다. 또한 부하분산 정책의 경우는 현재 영상처리객체의 처리특성과 처리서버의 성능을 고려할 수 있는 방안에 대한 보완이 요구된다. 즉, 현재 구현된 시스템의 부하분산 정책

은 단순히 round-robin 방식의 부하분산방식에 비해 객체의 상태 중에서 해당 객체가 사용중인지 아닌지에 대한 추가정보를 유지하여 해당 객체를 선택하는 방식과 해당 객체들을 처리하고 있는 서버에서 처리주인 작업의 수에 대한 정보를 바탕으로 부하분산을 시도하고 있다. 그러나 영상처리 작업은 처리하고자 하는 영상의 크기, 작업의 종류, 그리고 처리서버의 성능에 따라서 작업의 완료시간이 예측 가능한 특성을 지니므로 이를 이용하는 부하분산 정책에 대한 연구가 지속되어야 할 것이다.

5. 결 론

분산객체 시스템의 한 예인 MICS 시스템에서 요구되는 최적의 서비스 가능 객체에 대한 연결 서비스를 CORBA에서 제공하는 객체참조 메커니즘만으로는 제공할 수 없으므로 본 논문에서는 CORBA 환경하에서 좀더 유용하고, 고성능의 분산 컴퓨팅 환경을 제공할 수 있는 객체관리자라는 확장된 객체관리 시스템을 설계, 구현하였다. 구현된 객체관리자는 서비스 객체들의 등록과 결합허용 그리고 객체 모니터링 기능을 이용한 부하분산기능을 제공할 뿐 아니라 서비스 객체들의 참조를 관리하여 클라이언트 환경의 동적 재구축을 지원하며, 모든 서비스 객체에 대한 접근을 관리함으로써 임의의 사용자가 서비스 객체에 직접 접근 시 예상되는 오 동작을 방지할 수 있어 시스템의 안정적인 동작을 보장한다.

현재 제안된 객체관리자는 부하분산을 위해 서버의 작업계수기와 해당 객체의 사용가능여부를 나타내는 기본적인 데이터만 이용하고 있다. 따라서 요청된 작업의 예상 작업처리시간과 처리서버가 다양해지는 경우에 대비한 처리서버의 성능을 고려한 정교한 부하분산 방안에 대한 연구가 필요하다. 또한 모든 요청이 객체관리자를 통해 이루어지므로 처리서버가 많아지는 경우 이를 효과적으로 관리할 수 있는 방안에 대한 연구가 필요할 것이다. 그리고 지금은 단순히 모든 처리객체를 각 처리서버에 배치하였지만 이를 용도별로, 사용빈도별로 배치하는 방법에 대한 연구가 필요할 것이다. 마지막으로 현재의 시스템에는 객체관리자에게 모든 관리기능을 부여하고 있는데 시스템 관리의 용이성을 위해 객체관리자를 원격관리 할 수 있는 관리도구의 개발도 이루어져야 할 것이다.

참 고 문 헌

[1] 강재효, 서재현, 김상균, "CORBA 기반의 분산 객체 관리 모델 구현", 인제대학교 인제논총 제14권 1호, pp.923~934, Oct. 1998.

[2] 이한수, 류기열 "그룹모델을 이용한 CORBA Implementation Repository 결합허용기법" 한국정보처리학회 추계 학술발표논문집 제6권 제2호, PL-1~PL-6, 1999.10.

[3] Jon Siegel, "CORBA Fundamentals and Programming", Wiley, 1996.

[4] Robert Orfali, Dan Harkey, "Client/Server Programming with Java and CORBA", Wiley. 1997.

[5] Visigenic, ""Distributed Object Computing : A Challenge for the '90s", CORBA White Paper, Feb. 1997.

[6] 강재효, 김정현, 김상균 등, "CORBA 기반의 의료영상 회의 시스템 구현" 99봄 학술발표논문집, 한국정보과학회, 제26권 1호, pp.340~342, Apr. 1999.

[7] 김정현, 강재효, 성병우 등, "의료영상 회의시스템을 위한 서버 시스템의 요구분석 및 설계" 98 추계학술발표논문집, 한국멀티미디어학회, pp.15~19, 1998.12.

[8] Kim JH, Kang JH, Kim SG, et al. "Histopathological Microscopic Image Conference System Based on Web" 6th ESACP Congress Abstract 1999 A082, Apr. 1999.

[9] OMG, "CORBA services : Common Object Services Specification", OMG Document 97-12-02.

[10] OMG, "CORBA Specification v2.2", OMG Document, Feb, 1998.

[11] Bearman, M., "Tutorial on ODP Trading Function", 1997. Available HTTP : http://www.dstc.edu.au Directory : AU/research_news/odp/trader/trtute File : trtute.html.

[12] Belaid, D., "Dynamic Management of CORBA Trader Federation", In 4th USENIX Confer-

ence on Object-Oriented Technologies and Systems(COOTS), Santa Fe, New Mexico, Apr. 1998.

[13] 이해평, 김원태, 안광선, "개선된 코바 객체 트레이더 서비스의 설계 및 구현", 한국정보처리학회 추계 학술발표논문집 제6권 제2호, DIST-106~DIST-111, Oct. 1999.

[14] "What is Push?", 1997, Available HTTP: http://www.whatis.com/push.htm.

[15] Landis, S., Maffeis, S., "Building Reliable Distributed Systems with CORBA", submitted to Theory and Practice of Object Systems, John Wiley Publisher, NY, 1995.

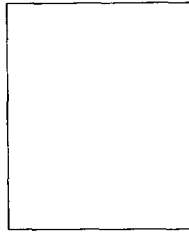
[16] Maffeis, S., "Adding Group Communication and Fault-Tolerance to CORBA", In Proceedings of the 1995 USENIX Conference on Object-Oriented Technologies, Monterey, CA, Aug. 1995.

[17] Silvano Maffeis, Alexey Vaysburd, Sophia Georgiakaki, "An Interoperable Middleware Infrastructure Supporting Object-Group Communication", SIGCOMM Workshop Position Paper.

[18] Silvano Maffeis, "A Fault-Tolerant CORBA Name Server", Department of Computer Science, Cornell University, 1996.

[19] Thoma J. Mowbray and Raphael C. Malveau, "CORBA Design Patterns", John Wiley & Sons, 1997.

박 세 명

1986년 2월 경북대학교 전자공학과(전산공학), 공학사
1985년 2월 경북대학교 전자공학과(전산공학), 공학석사
1994년 8월 경북대학교 전자공학과(전산공학), 공학박사
1990년~현재 인제대학교 정보컴퓨터공학부 부교수
관심분야 : 분산시스템, 분산데이터베이스, 컴퓨터네트워크

강 재 효

1998년 2월 인제대학교 전산학과, 이학사
2000년 2월 인제대학교 전산학과, 이학석사
2000년 3월~현재 에이블컴(정보기술팀)
관심분야 : 분산시스템, 컴퓨터보안, 홈네트워킹

김 상 균

1991년 경북대학교 통계학과(이학사)
1994년 경북대학교 대학원 컴퓨터공학과(공학석사)
1996년 경북대학교 대학원 컴퓨터공학과(공학박사)
1996년~현재 인제대학교 정보컴퓨터공학부 조교수
관심분야 : 정보처리

최 항 묵

1982년 2월 연세대학교 공학사
1987년 12월 University of Oklahoma, Computer Science, 석사
1996년 7월 University of Oklahoma, Computer Science, 박사
1997년~현재 인제대학교 정보컴퓨터공학부 조교수
관심분야 : 소프트웨어공학, 분산시스템, 전자상거래

최 흥 국

1988년 Linköping Universiy, Computer Engineering, Linköping, Sweden (공학사)
1990년 Linköping Universiy, Computer Engineering, Linköping, Sweden (공학석사)
1996년 Uppsala Universiy, Computer Image Analysis, Uppsala, Sweden(공학박사)
1996년~1997년 서울대학교 의공학연구소
1997년~현재 인제대학교 정보컴퓨터공학부 조교수
1997년~1999년 인제대학교 전산정보처장
1997년~현재 한국멀티미디어학회 논문지 편집위원
관심분야 : 멀티미디어, 컴퓨터그래픽스, 영상처리 및 분석